SEI Webinar Series:

Agile Development & Software Architecture – Crossing the Great Divide

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Nanette Brown
4/22/2010

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **22 APR 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE **Agile Development & Software Architecture - Crossing the Great Divide**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University ,Software Engineering Institute,Pittsburgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**60**	

Today's Presenter

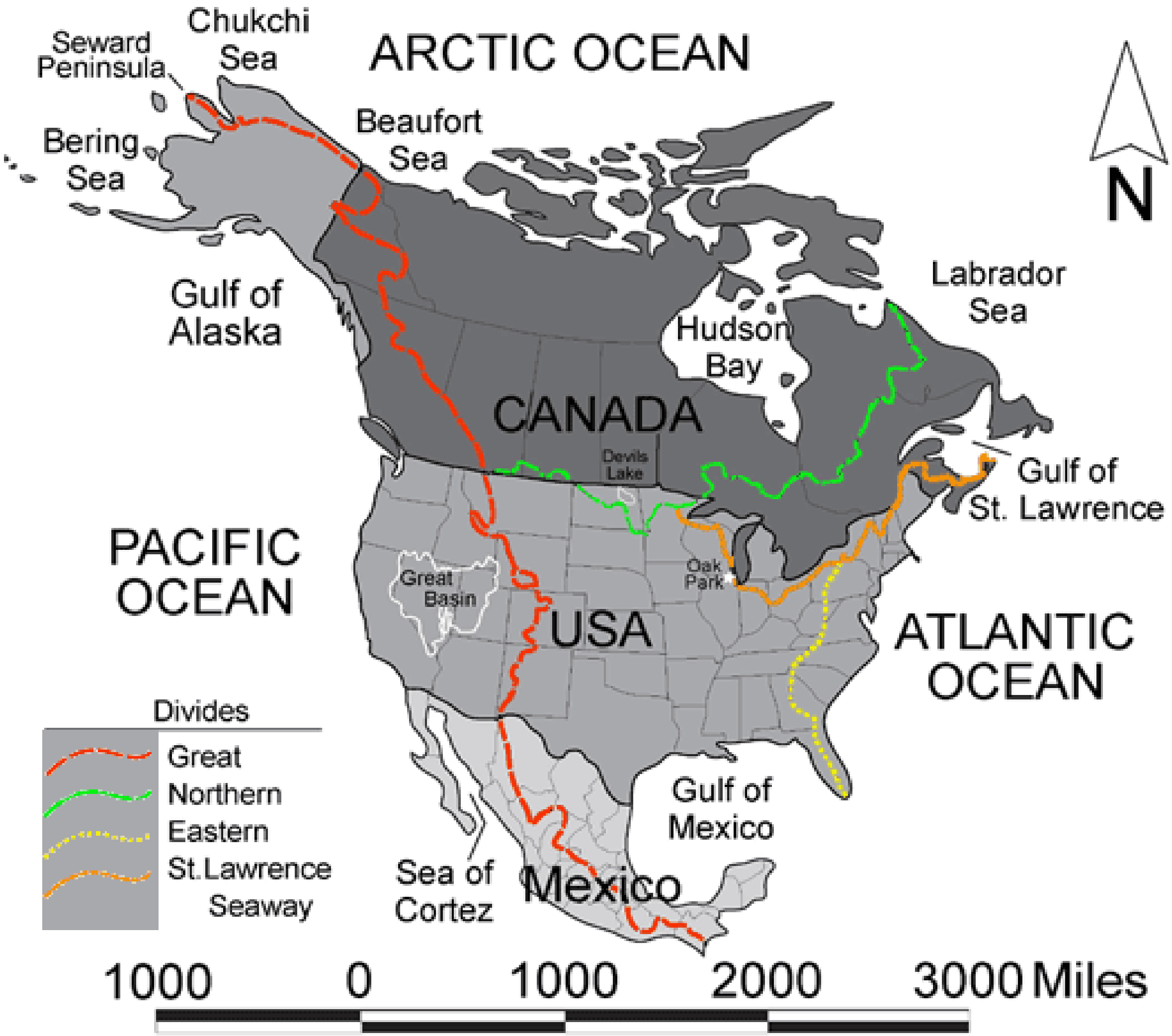
Nanette Brown is a Visiting Scientist with the Software Engineering Institute's Research, Technology, and System Solutions Program and is a Principal Consultant with NoteWell Consulting. She is currently engaged in an SEI Research Project on "Communicating the Value of Architecting within Agile Development" as well as other activities focusing on architecture within an Agile context.

Previously, Nanette worked at Pitney Bowes Inc., most recently as Director of Architecture and Quality Management, where she was responsible for design and implementation of a customized SDLC that blended RUP and Agile practices. Nanette has presented at multiple industry conferences including SD Best Practices and Project World and the World Conference of Business Analysts on topics such as Facilitated Iteration Planning and the SEI scenario-based approach to specify quality attributes.

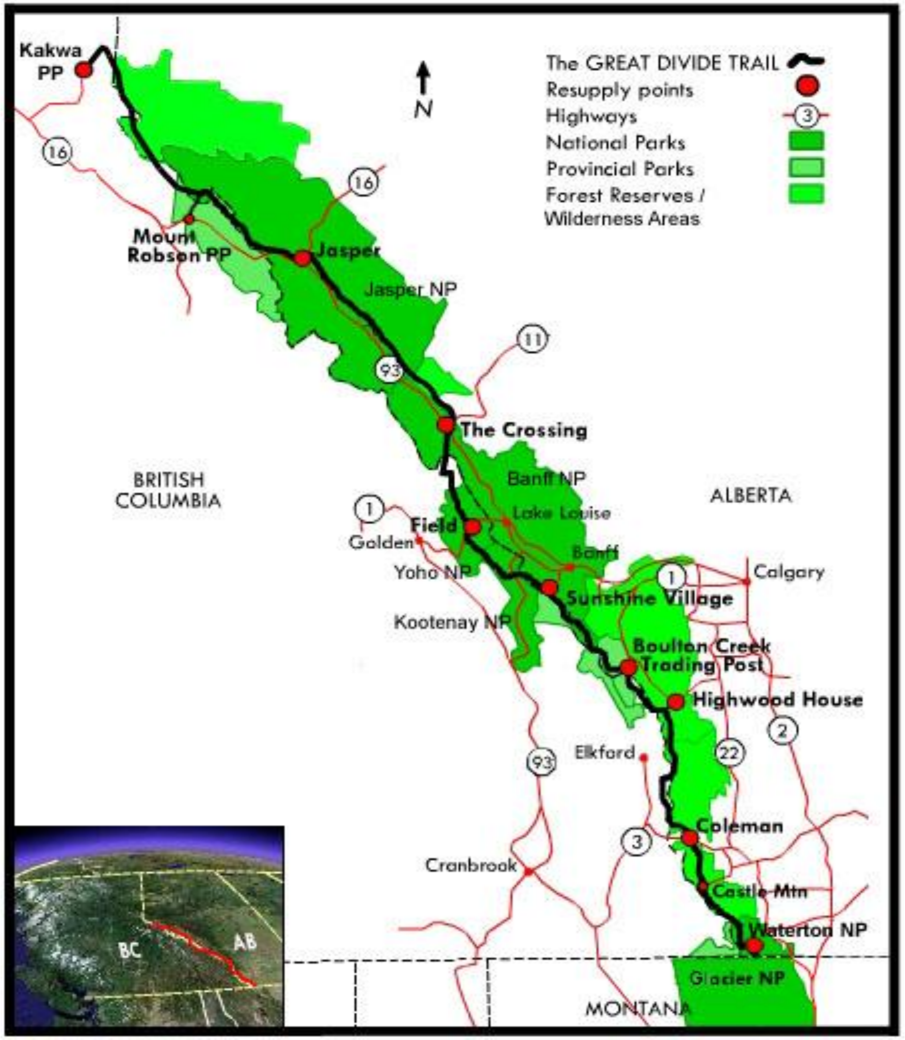
Polling Question #1

How did you hear about this webinar?

1. Social Media (i.e., LinkedIn, Twitter)
2. SEI Website
3. SEI Member Bulletin
4. Email invitation from the SEI
5. Website with webinar calendar (i.e., www.webinar-directory.com)



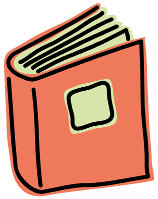
http://en.wikipedia.org/wiki/Continental_Divide_of_the_Americas



http://en.wikipedia.org/wiki/Great_Divide_Trail

“The **Great Divide Trail**, or GDT, is a wilderness hiking trail in the Canadian Rockies. The trail closely follows the Continental Divide between Alberta and British Columbia, crossing the divide no fewer than 30 times. (…)

The GDT is not officially recognized by Parks Canada and therefore is not signed and not always even an actual trail, <u>sometimes merely a wilderness route</u>.”*



* http://en.wikipedia.org/wiki/Great_Divide_Trail

Crossing the Great Divide

Polling Question #2

Are you currently using Agile Development practices within your organization?

1. Yes
2. No
3. Not sure

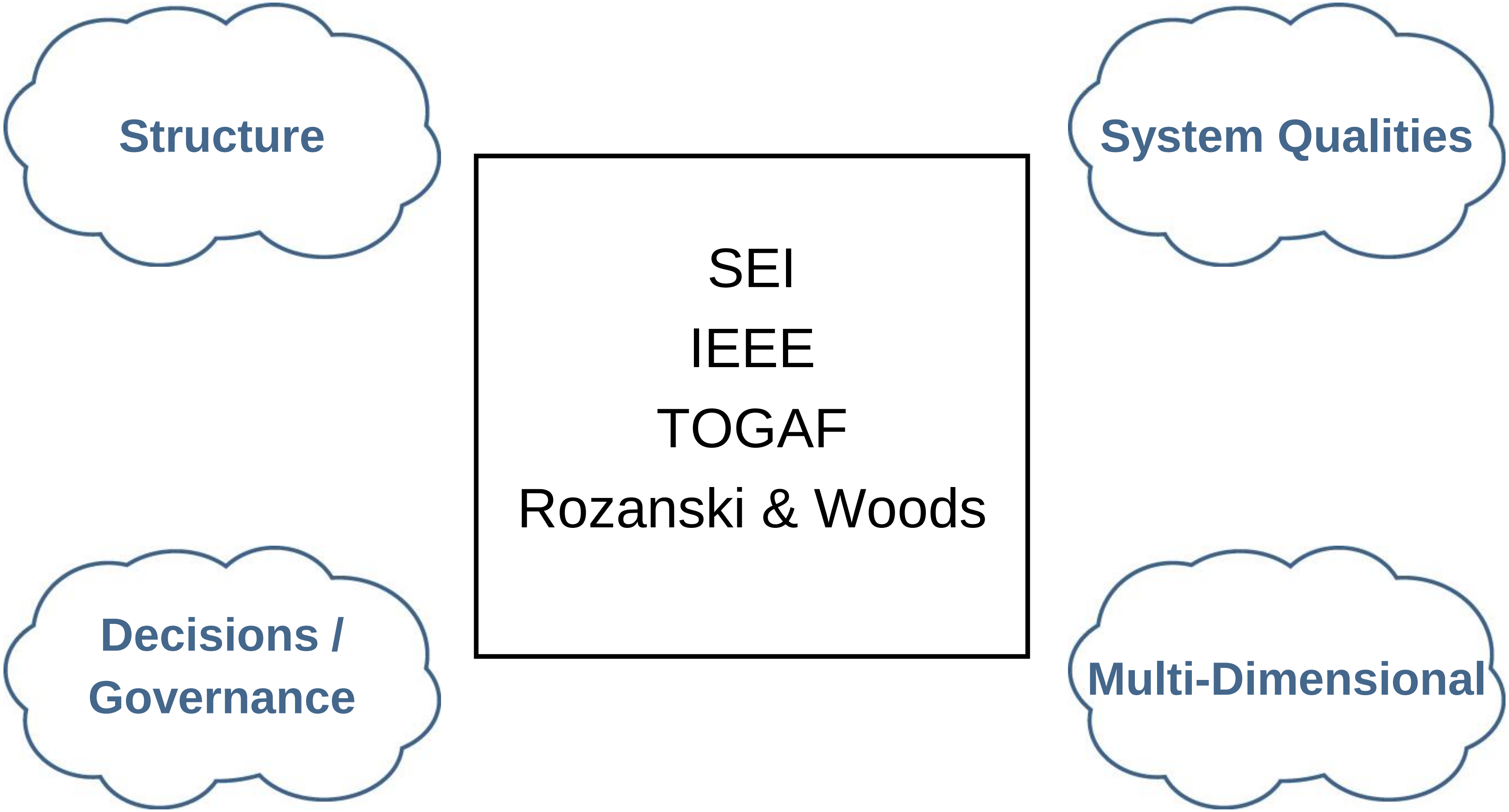
Polling Question #3

Do you think that Agile Development and Software Architecture are

1. In conflict
2. Complementary
3. I’m not sure

Scouting the Terrain

What is Architecture?
A Thematic Analysis



Structure

System Qualities

SEI

IEEE

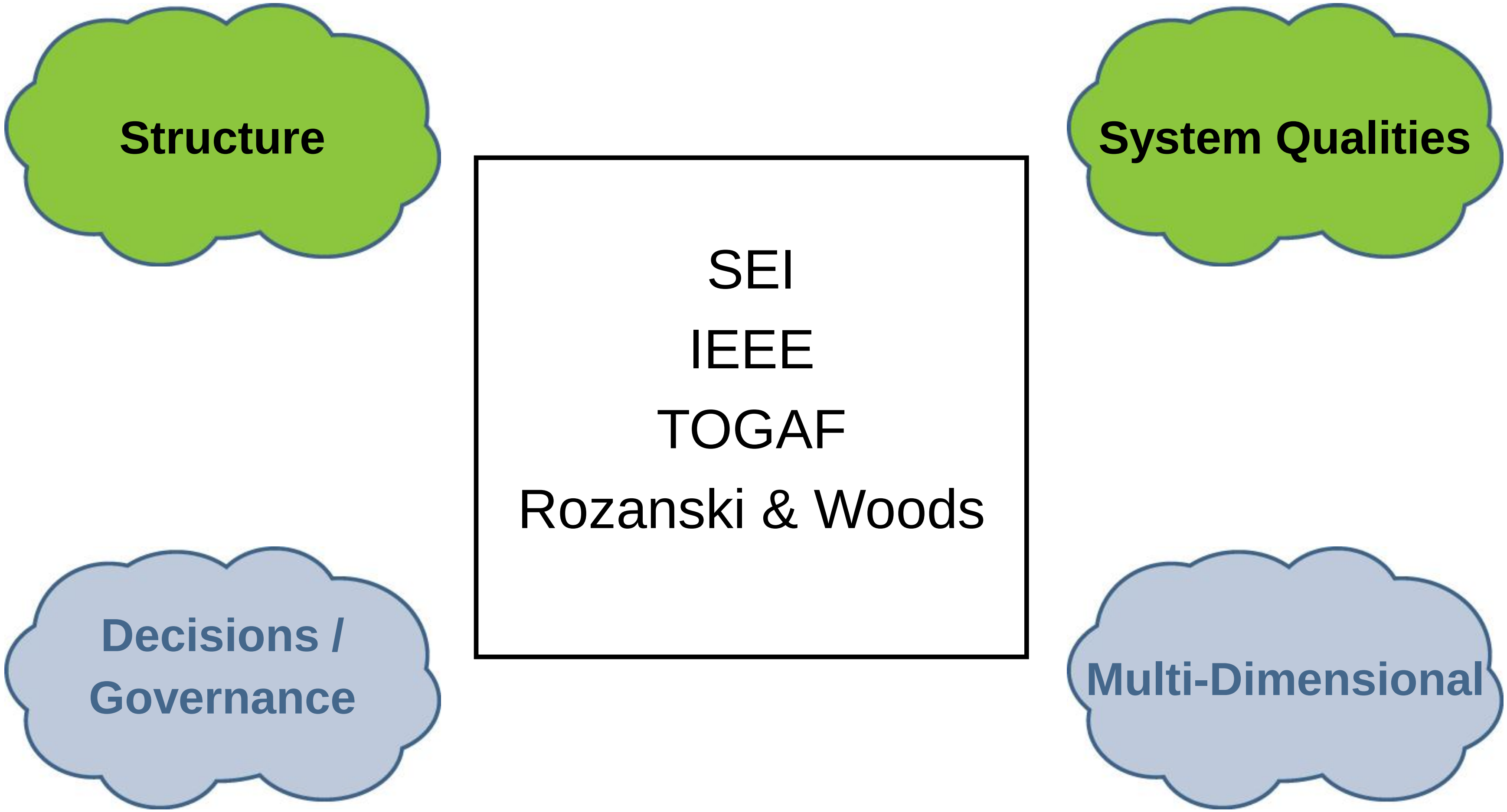
TOGAF

Rozanski & Woods

Decisions / Governance

Multi-Dimensional

Architectural Themes



Structure

System Qualities

SEI

IEEE

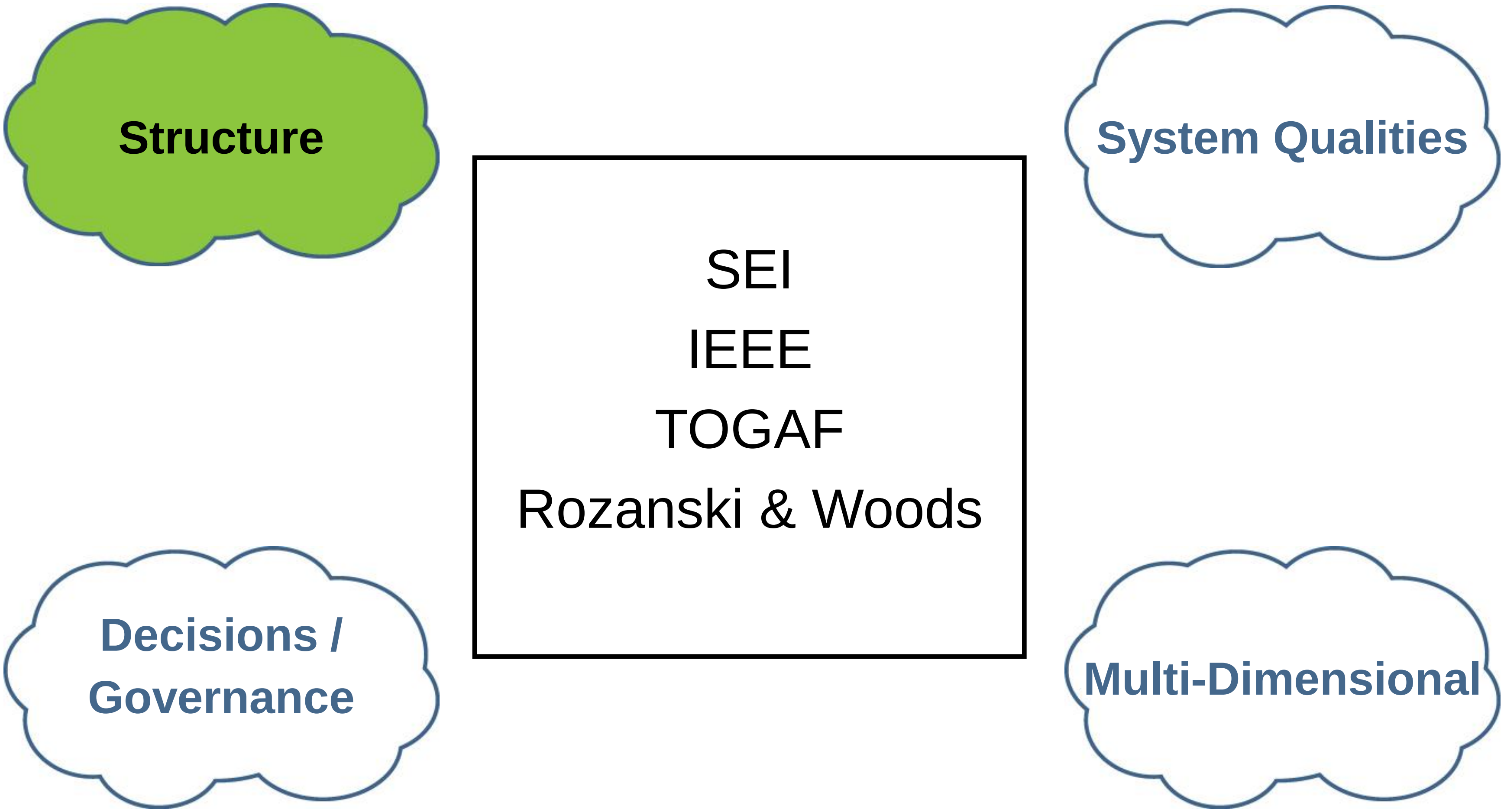
TOGAF

Rozanski & Woods

Decisions / Governance

Multi-Dimensional

Architectural Themes



Structure

System Qualities

SEI

IEEE

TOGAF

Rozanski & Woods

Decisions / Governance

Multi-Dimensional

Structural Theme – Agreement

Both Agilists and Architects agree that that the structure of a system (i.e., the system’s decomposition into components and component inter-relationships) is a real and significant concern.

Structural Theme – Debate

Debate between the Agile and Architectural communities focuses on the malleability of that structure and the extent to which it should be pre-defined or allowed to emerge throughout the course of development.

Structural Theme – Resolution

Context is key

- Project size – features, code, team size
- Product criticality

Alistair Cockburn
Crystal Family of Methodologies

Criticality

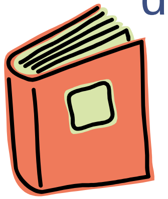
Life	L6	L20	L40	L80
Essential Money	E6	E20	E40	E80
Discretionary Money	D6	D20	D40	D80
Comfort	C6	C20	C40	C80

Team Size

Structural Theme - Resolution

Context is key

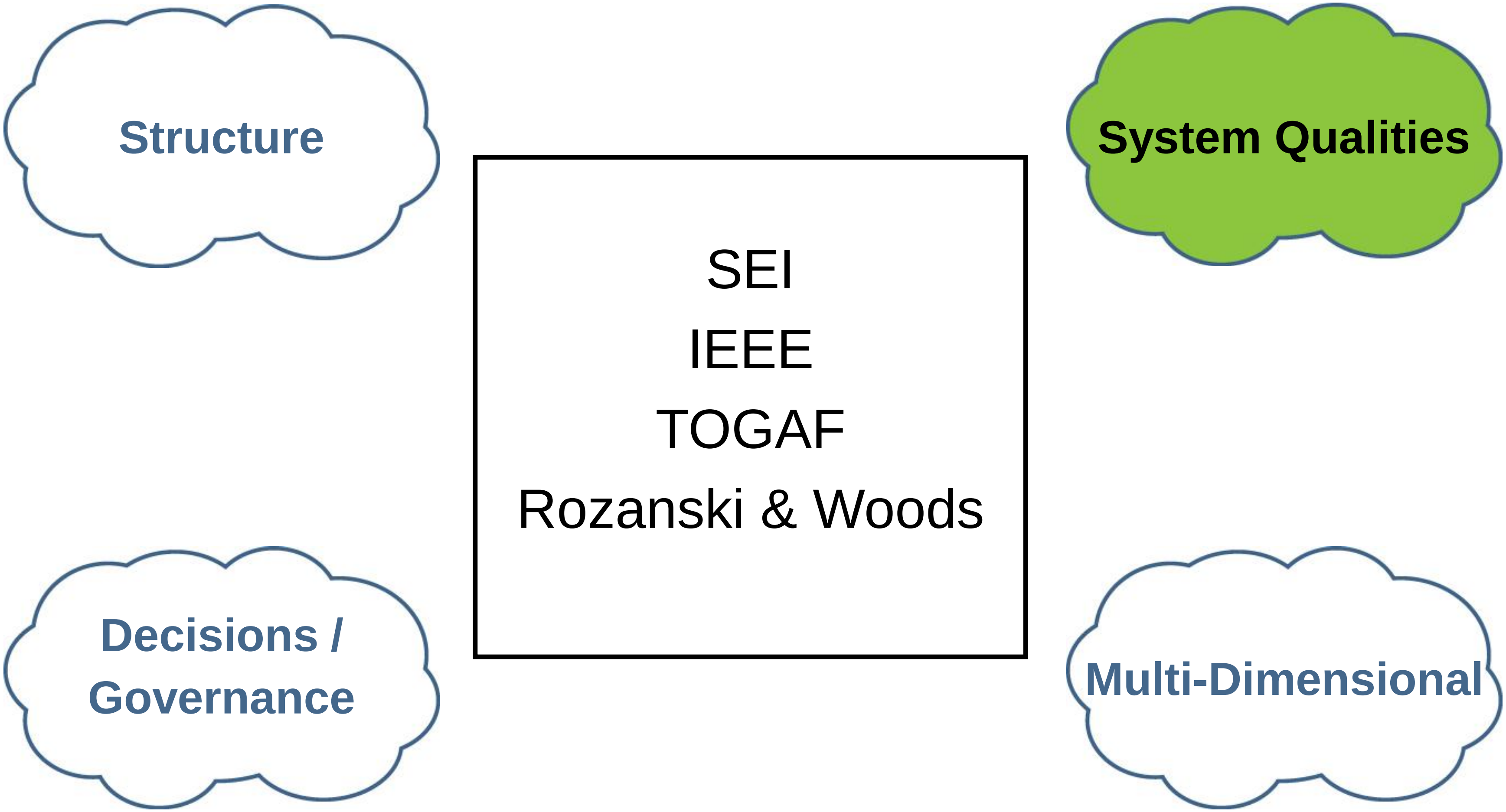
- Organizational constraints
  - – Geographical distribution
  - – Culture – tolerance for ambiguity and risk, trust
- Discovery and innovation
  - – New, unknown emerging market or well-established domain?
  - – Maturity of technology / organizational technology experience base
- Technology
  - – Embedded System vs. Enterprise Architecture
  - – Flexible commercial product framework or “close to the metal” development environment



Philippe Kruchten Blog Post – “The Context of Software Development”

http://pkruchten.wordpress.com/2009/07/22/the-context-of-software-development/

Architectural Themes



Structure

System Qualities

SEI

IEEE

TOGAF

Rozanski & Woods

Decisions / Governance

Multi-Dimensional

System Qualities Theme

Quality attributes are typically referred to as non-functional requirements. They represent system characteristics (i.e., qualities of a system) such as performance, availability, scalability, modifiability, and testability.

System Qualities Theme

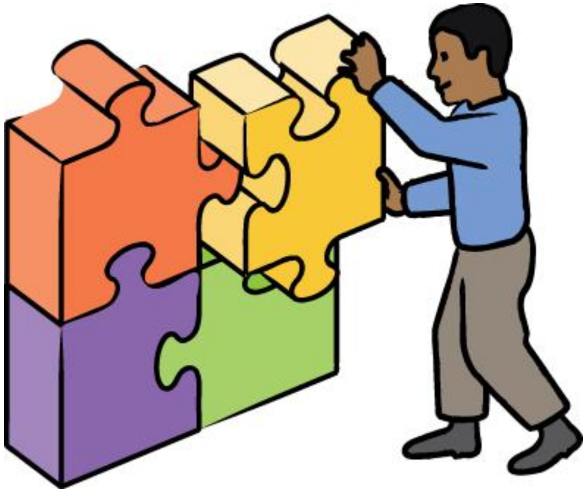
Quality attributes may be classified by their primary stakeholders:

- End-user stakeholders
- Development stakeholders
- Delivery & support stakeholders

Examples of Quality Attributes

End-User Stakeholders	• Reliability • Performance • Availability • Security • Usability
Development Stakeholders	• Modifiability • Testability • Portability • Reusability
Delivery & Support Stakeholders	• Installability • Diagnosability • Per-Click-Sales-Ability • SaaS-Ability • Auditability • Reconcilability

Eliciting / Expressing Quality Attributes

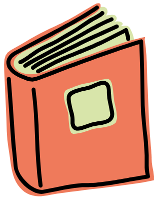


Quality Attribute Scenarios

Stimulus

Environment

Response



Bass, Clements, Kazman – “Software Architecture in Practice”

Eliciting / Expressing Quality Attributes

“Forgotten Stakeholder” Stories

As a <Stakeholder>, I want <some goal>

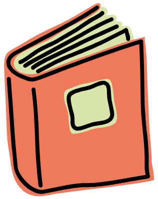
so that <some reason>”

Acceptance Tests

Given preconditions (Environment)

When actions or triggers (Stimulus)

Then consequences (Response)



Mike Cohn – “User Stories Applied”

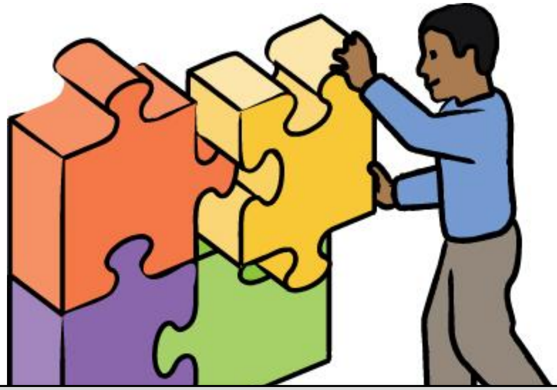
Dan North – “Introducing BDD” - http://blog.dannorth.net/introducing-bdd/

Polling Question #4

How are quality attributes (or non-functional requirements) elicited / expressed in your organization?

1. Declarative statements (the system shall …)
2. Scenarios or stories
3. Test Cases
4. Not much attention explicitly paid to quality attributes
5. Other

When Should Quality Attributes be Addressed?



"The quality attributes of any nontrivial system are determined by its architecture."[1]

"Make it work, then make it faster."[2]

1 Clements, Kazman, Klein – "Evaluating Software Architectures", p. 109

2 Cohn – "Agile Estimating and Planning" p. 126 – reference to "The Elements of Programming Style" by Kernighan and Plauger

When Should Quality Attributes be Addressed?

Context is Key!

Tools for Navigation

Agile & Architecture in Release Planning

Release Planning is a critical life-cycle practice.

Release Planning forces choices that bring into focus issues of cost and value, current needs, and future potential.

Agile & Architecture in Release Planning

How to integrate architectural considerations into Agile release planning?

How to make the release planning for architecture more Agile?

Life Cycle and Release Planning

Life cycle has a major influence on the way in which architecture is addressed during release planning.

Architecture within the SDLC

Agile – Inter-weave architectural implementation with the implementation of stakeholder stories.

RUP – Design, code, and test architecture during Elaboration Phase iterations by focusing on architecturally significant requirements. Non-architecturally significant requirements are defined, implemented, and tested in the Construction Phase.

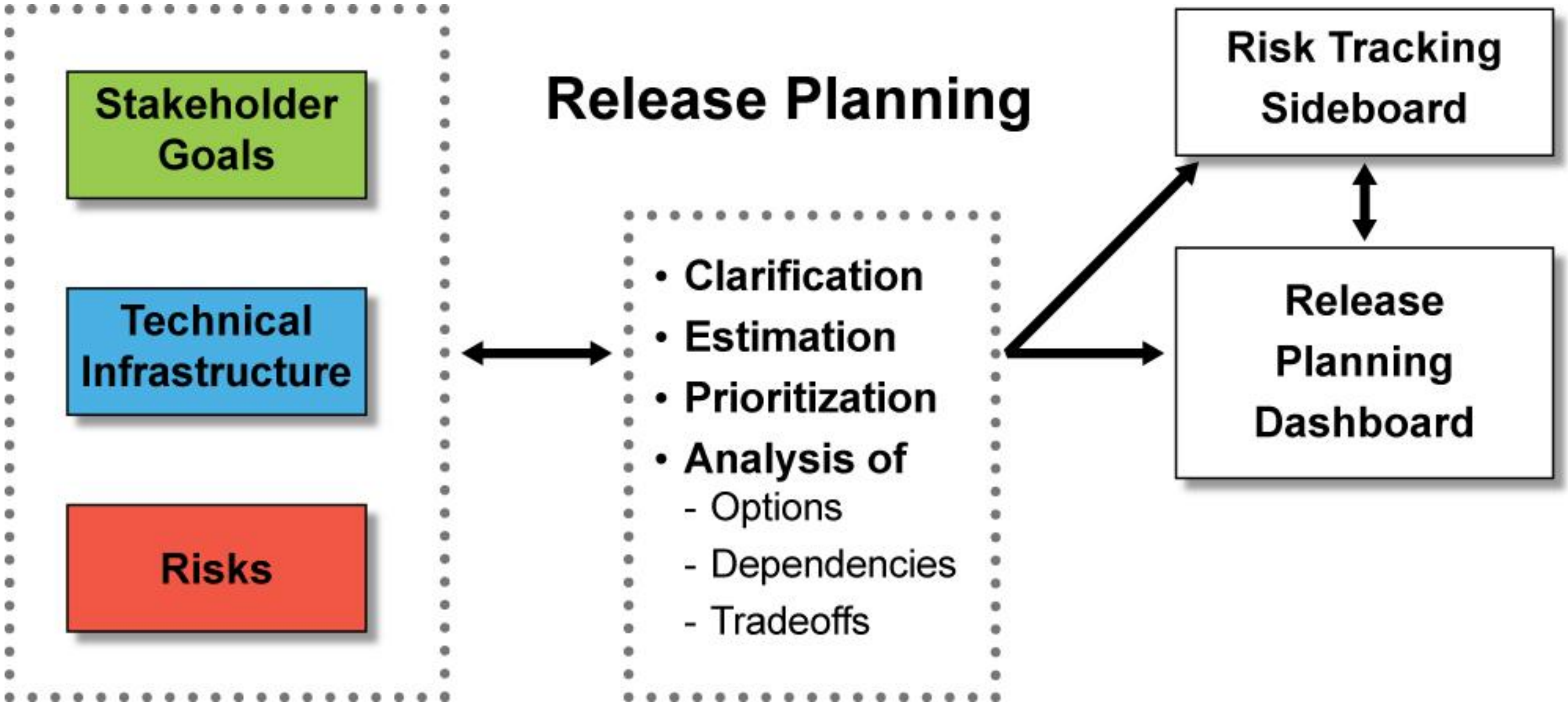
Waterfall – Define all requirements, complete all architecture and design, complete all coding, and perform all test activities.

Polling Question #5

Which of these descriptions most closely matches development practices at your organization?

1. Waterfall
2. RUP
3. Agile
4. None of the Above

Architecture in Agile Release Planning



Release Planning Inputs

Stakeholder Goals

- Include
  - Functionality
  - Quality attributes
  - Constraints
- Expressed by
  - End users
  - Development team
  - Delivery and support team
- Expressed as
  - Stories
  - Acceptance test cases

Release Planning Inputs

Technical Infrastructure

- Includes
  - Architectural implementation / enhancements
  - Technical research / technology selection
  - Code-level and architectural refactoring
  - Technical debt – incursion and reduction
- Expressed by
  - Development team
- Expressed as
  - Stories
  - Design spikes
  - Tasks
  - Technical models / sketches / documentation

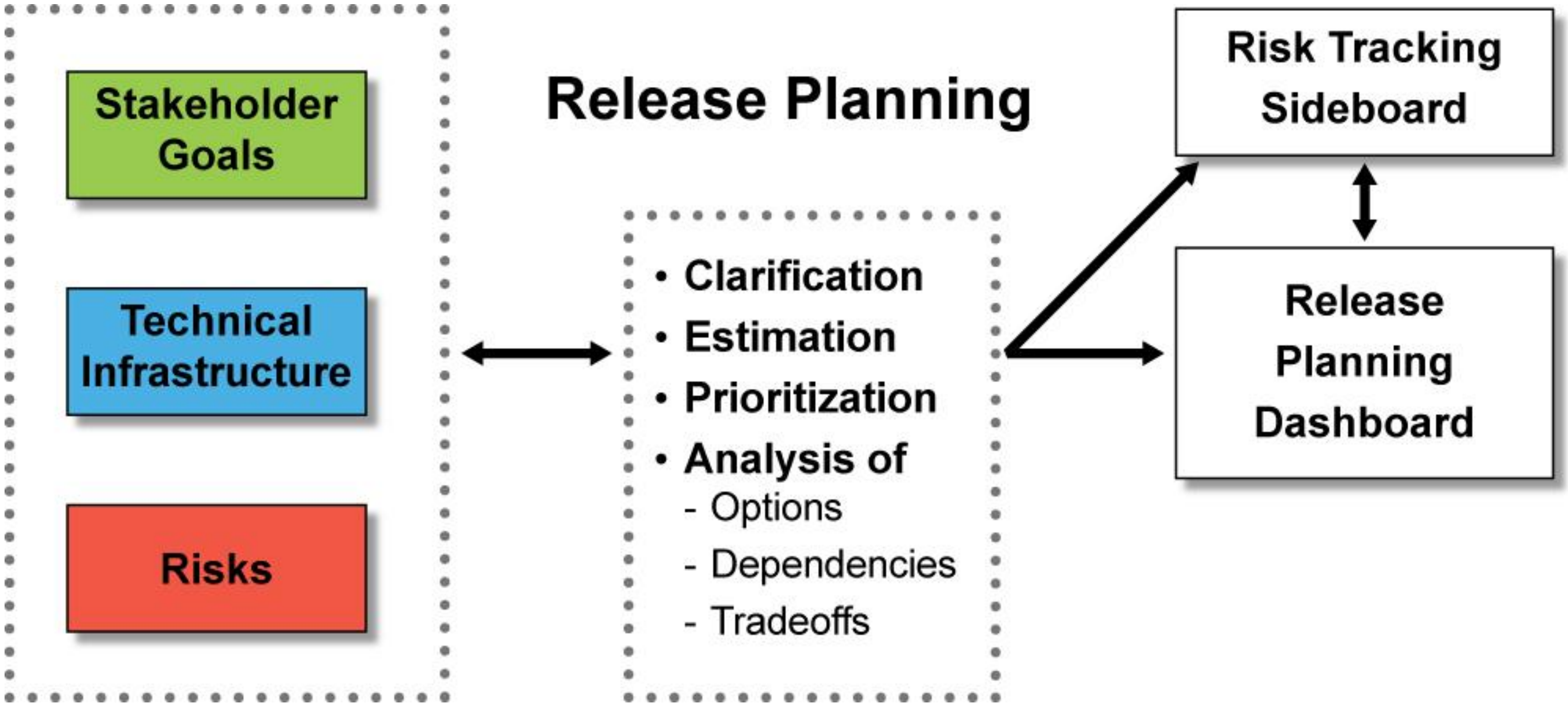
Release Planning Inputs

Risks*

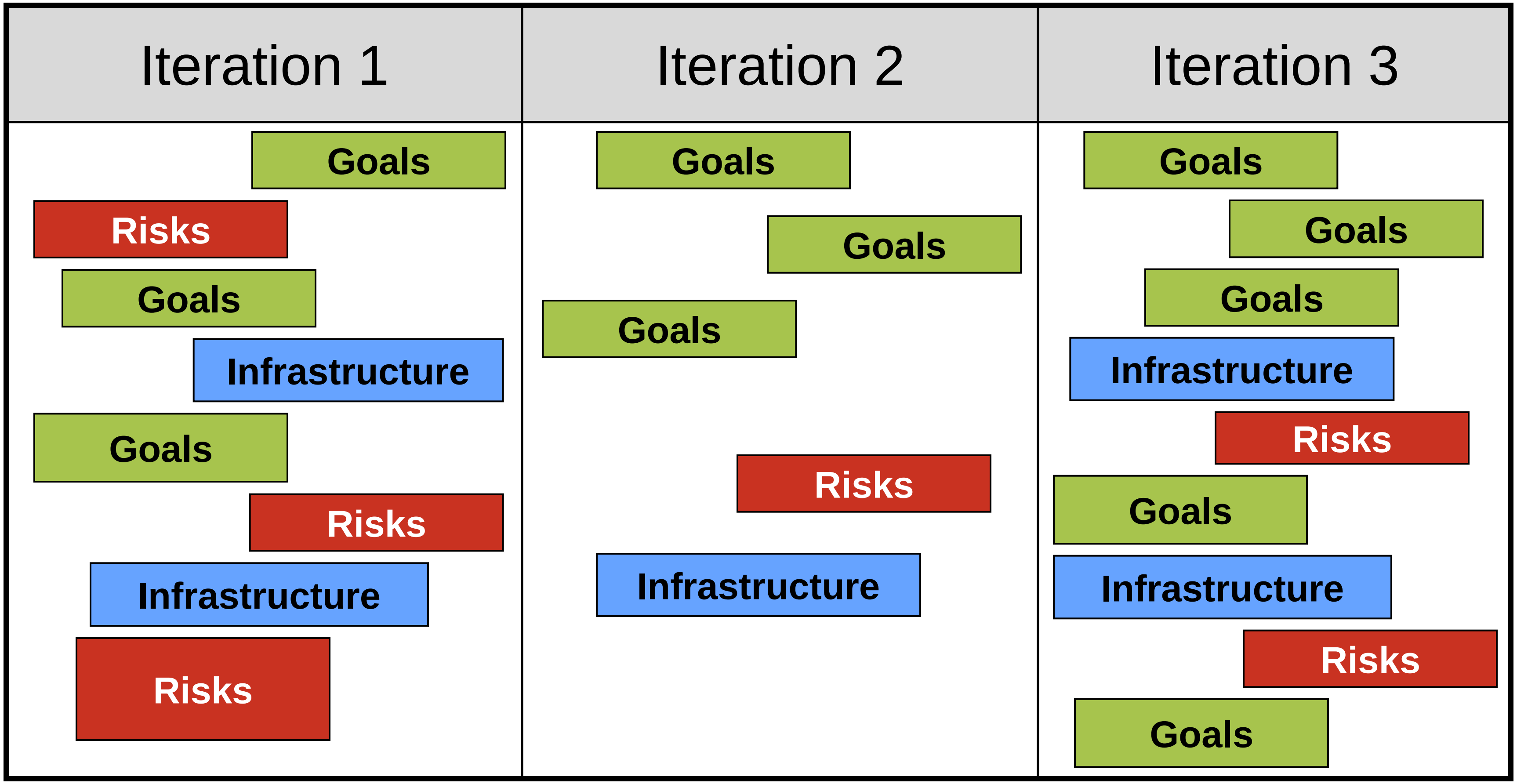
- Include
  - Technical risks
  - Market risks
  - End-user acceptance risks
  - Deployment risk, etc.
- Expressed by
  - End users
  - Development team
  - Delivery and support team
- Expressed as
  - Stories – “As a <Stakeholder> I want <risk mitigation action> so that <risk mitigation result>”

***Risks that influence release planning outcomes**

Architecture in Agile Release Planning

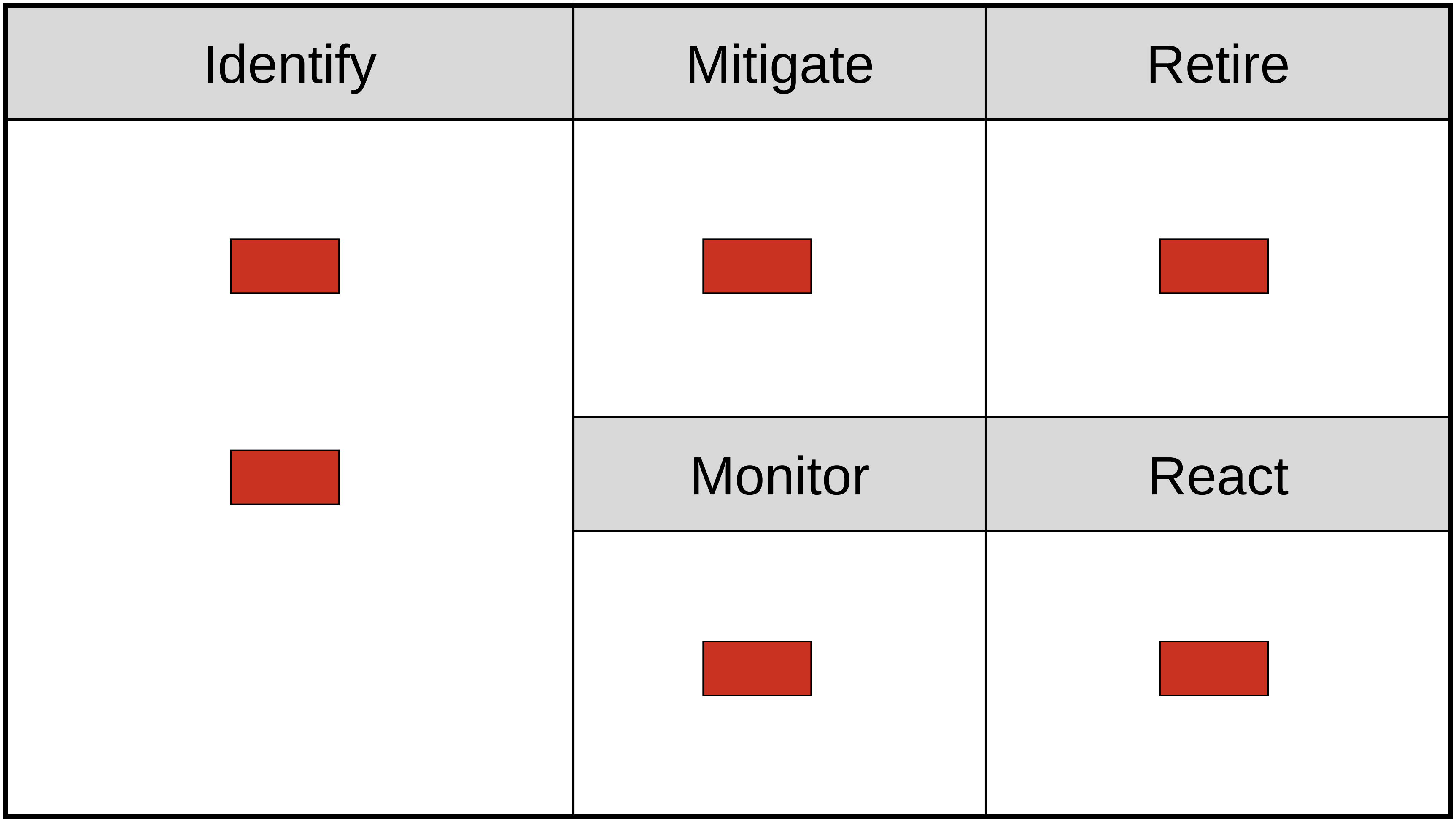


Release Planning Dashboard



Iteration 1	Iteration 2	Iteration 3
Goals	Goals	Goals
Risks	Goals	Goals
Goals	Goals	Goals
Infrastructure	Risks	Infrastructure
Goals	Infrastructure	Risks
Risks		Goals
Infrastructure		Infrastructure
Risks		Risks
		Goals

Risk Tracking Sideboard



Identify

Mitigate

Retire

Monitor

React

The Question of Real Options

Should I make an architectural investment in anticipation of a future need for a given story? *(i.e., functional or quality attribute story)*

How likely is it that the future need will arise?

Is there an architectural investment that I can make now that will reduce the future cost to implement the story?

If so, what is the cost of this architectural investment?

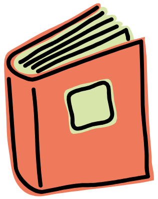
(e.g., cost to implement, opportunity cost, etc.)

What are the relative economics of meeting the future need with or without having made the prior architectural investment?

(e.g. , relative cost and time to implement with or without prior architectural investment, potential opportunity cost from delay in meeting the future stakeholder need, etc.)

Technical Debt

Technical Debt is a metaphor developed by Ward Cunningham as a means of explaining the need for refactoring to non-technical product stakeholders.



Cunningham, *The WyCash Portfolio Management System*. OOPSLA '92 Experience Report,

http://c2.com/doc/oopsla92.html

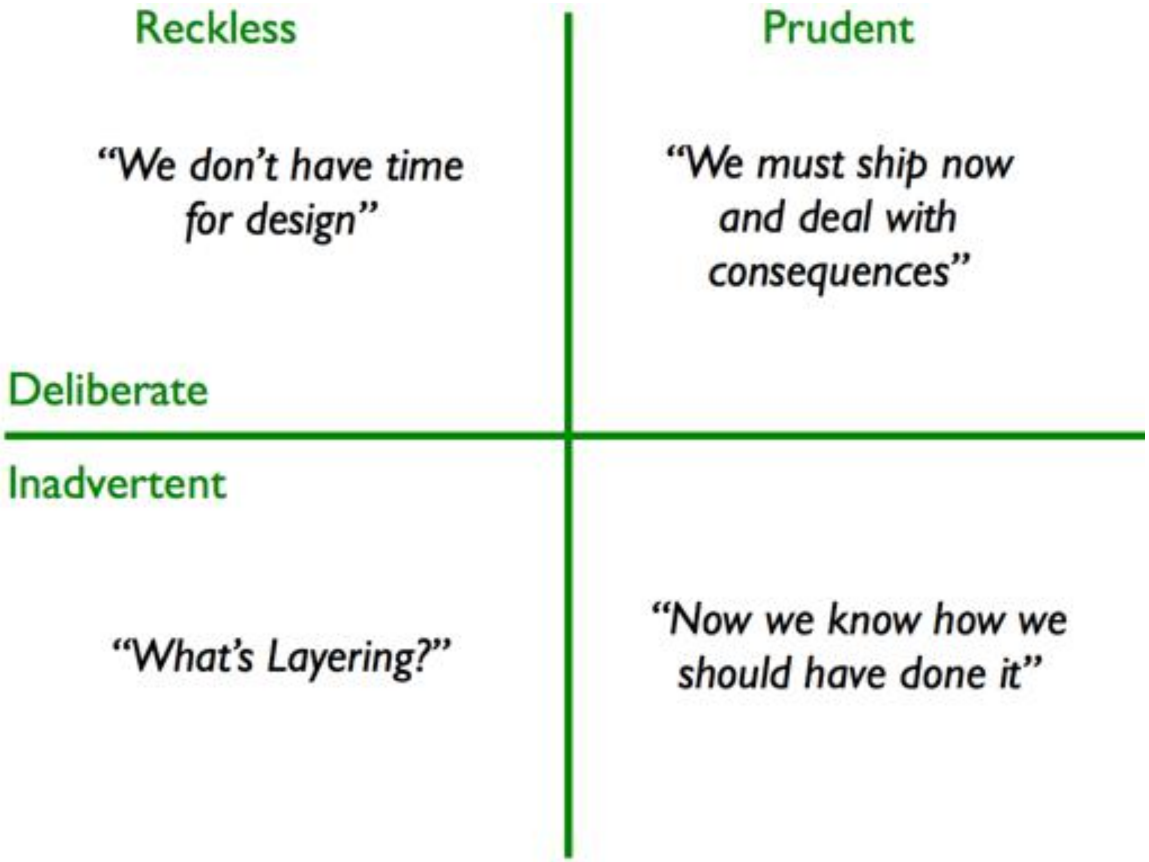
Technical Debt

Releasing a system with suboptimal architecture, design and / or code burdens the development organization with debt.

The interest payments associated with the debt cause future system enhancements to require increased time and effort.

If re-factoring techniques are not used to pay down the debt, debt can continue to accumulate to the point where enhancement activities grind to a halt, resulting in metaphorical (and potentially literal) bankruptcy.

Martin Fowler’s Taxonomy of Technical Debt



Reckless

Prudent

“We don’t have time for design”

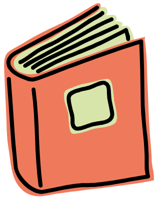
“We must ship now and deal with consequences”

Deliberate

Inadvertent

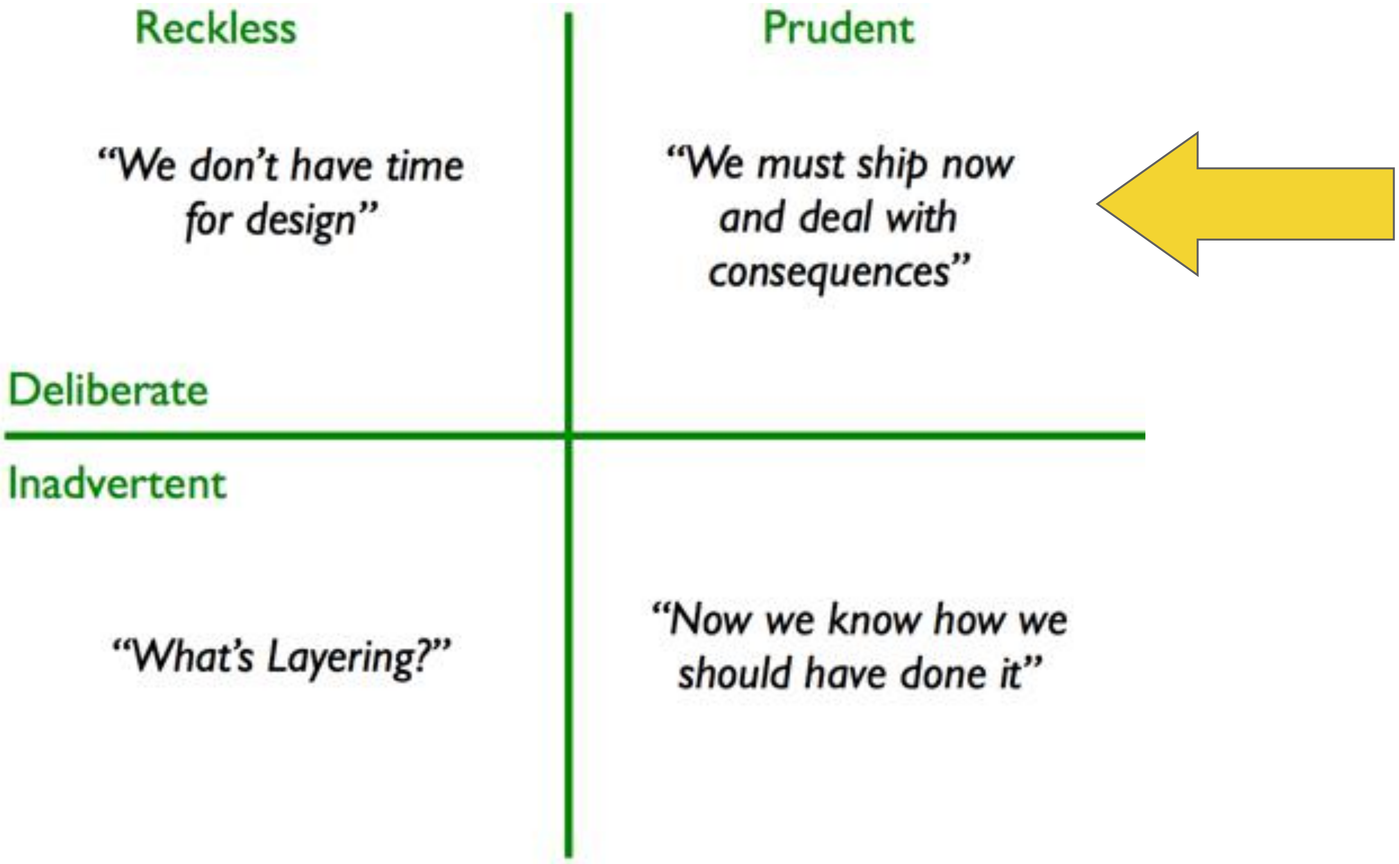
“What’s Layering?”

“Now we know how we should have done it”



http://martinfowler.com/bliki/TechnicalDebtQuadrant.html

Real Options and Technical Debt



Reckless

Prudent

"We don't have time for design"

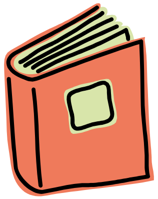
"We must ship now and deal with consequences"

Deliberate

Inadvertent

"What's Layering?"

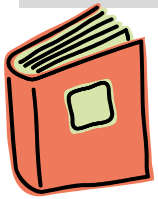
"Now we know how we should have done it"



http://martinfowler.com/bliki/TechnicalDebtQuadrant.html

Considering Dependencies in Agile Release Planning

Dependencies between stories & supporting architectural elements	Understanding the dependencies between stories and architectural elements enables staged implementation of technical infrastructure in support of achieving stakeholder value.
Dependencies between architectural elements	Low-dependency architectures are a critical enabler for scaling-up agile development.[1]
Dependencies between stories	High-value stories may require the implementation of lower-value stories as precursors.[2]



[1] Mary and Tom Poppendieck – “Leading Lean Software Development”

[2] Mark Denne, Jane Cleland-Huand – “Software by Numbers”

Where Do We Go From Here?

Focus on:

Design of agile practices for

- Real options analysis
- Quantifying architectural value
- Dependency analysis
- Managing technical debt
  - Incursion (real options analysis)
  - Reduction (enhancement of technical infrastructure)

We Hope You Will Join Us On the Journey

We Welcome

- Comments
- Questions
- Critiques
- Ideas
- Anecdotes
- Experience Reports
- Collaboration Opportunities

http://saturnnetwork.wordpress.com/

Nanette Brown
nb@sei.cmu.edu

Discuss Agile Development and Software Architecture further at SATURN 2010

Keynotes and invited talks

Jim Highsmith	*Architects: Accelerators or Anchors to Organizational Agility?*
Wayne Longcore	*Managing scale and agility: Transformational Architecture for the Smart Grid*
Philippe Kruchten	*Software architecture and agility: a clash of two cultures?*

Discuss Agile Development and Software Architecture further at SATURN 2010

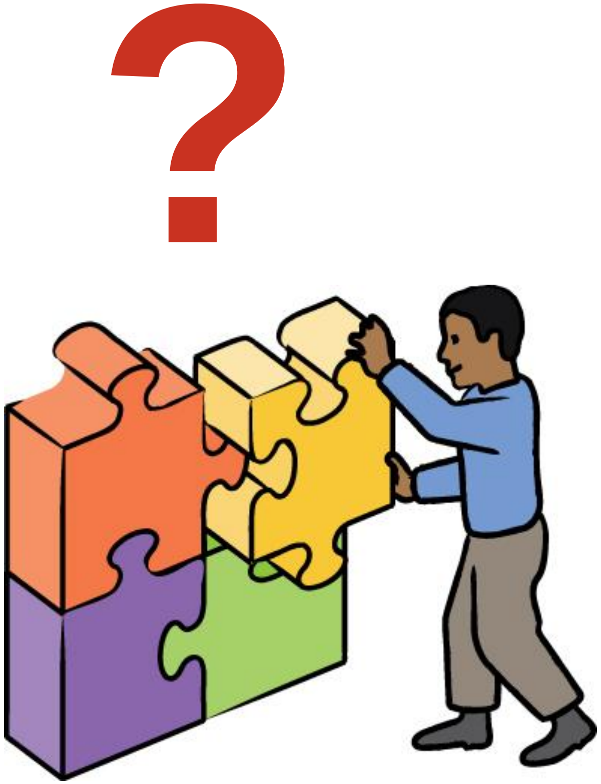
Presentations

Architecture and Agile, Friends or Enemies?
Ger Schoeber, Sioux Embedded Systems B.V.

Designing and Building Large-Scale Systems in an Agile World
Stevie Borne, Dave Henricksen, Thomson Reuters,

Agile Architecting: Using Agile Principles to Agilitize the Architecting Process
Amine Chigani, Virginia Tech

Agile Architect - Integrating Enterprise Architecture into Agile and Lean Software Development Environments
Srini Penchikala, InfoQ



Questions??

Comments!!

JOIN THE CONVERSATION

SATURN → BLOG

JOIN THE DISCUSSION →

SATURN → NETWORK ON Linked in

The SEI Architecture Technology User Network (SATURN) is a professional network of software, systems, and enterprise architects from around the world, representing industry, academia, and government.

SATURN 2010

Sixth Annual SEI Architecture Technology User Network Conference

ARCHITECTING FOR CHANGE

May 17-21, 2010
Minneapolis, Minnesota

REGISTER NOW → www.sei.cmu.edu/saturn/2010

SEPGSM 2010
NORTH AMERICA
Perform at a Higher Level

SEPG Europe 2010

28 June - 1 July, 2010
Porto, Portugal

www.sei.cmu.edu/sepg/europe/2010/

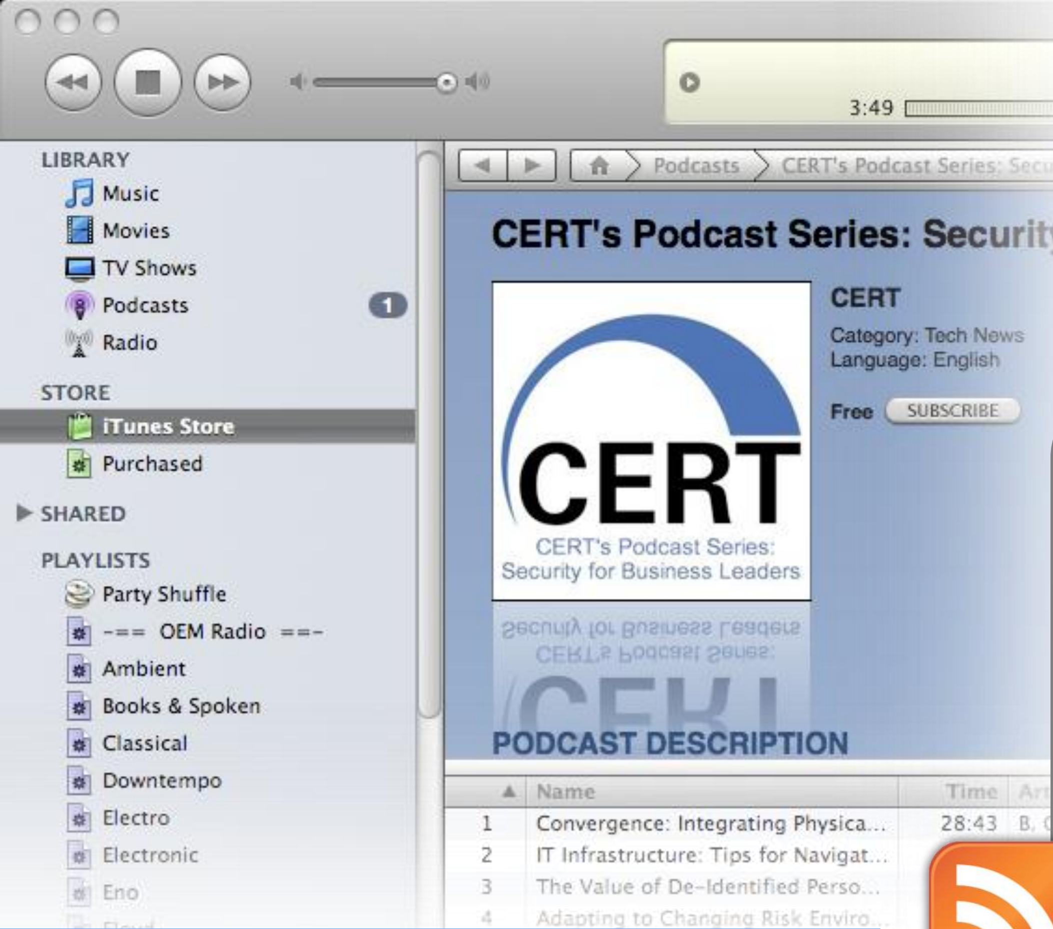
SEPG is the premier, global conference series on software and systems process management

Become an SEI Member!

▶ www.sei.cmu.edu/membership

SEI Training

▶ www.sei.cmu.edu/training



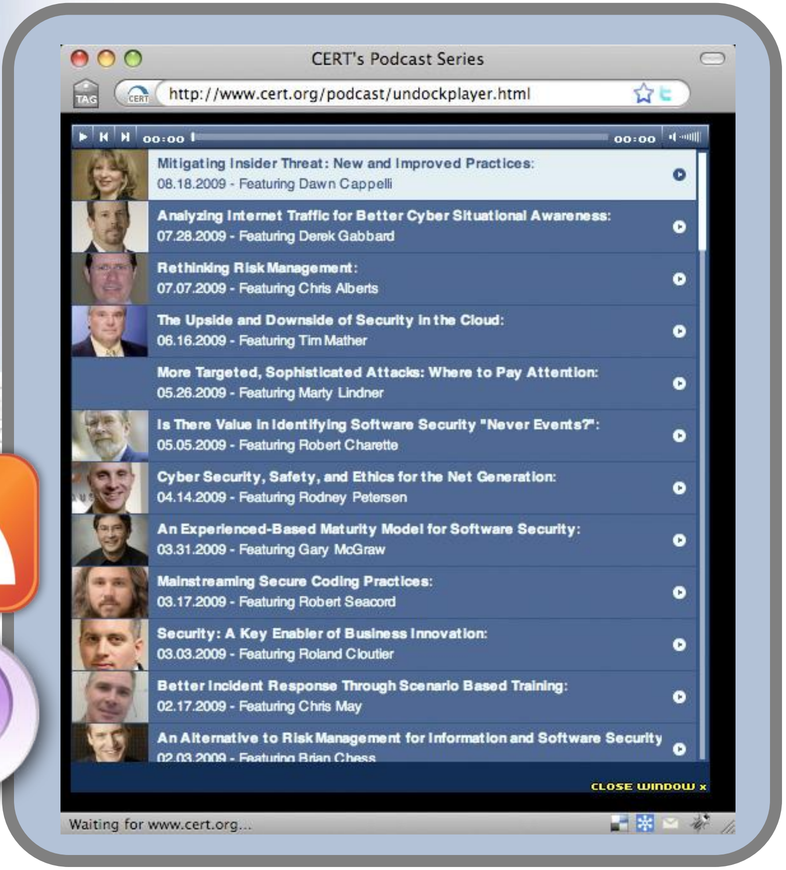
CERT's Podcast Series: Security for B

CERT

CERT's Podcast Series: Security for Business Leaders

CERT
Category: Tech News
Language: English

	Name	Time	Art
1	Convergence: Integrating Physica…	28:43	B. C
2	IT Infrastructure: Tips for Navigat…		
3	The Value of De-Identified Perso…		
4	Adapting to Changing Risk Enviro…		



CERT's Podcast Series

http://www.cert.org/podcast/undockplayer.html

Mitigating Insider Threat: New and Improved Practices:
08.18.2009 - Featuring Dawn Cappelli

Analyzing Internet Traffic for Better Cyber Situational Awareness:
07.28.2009 - Featuring Derek Gabbard

Rethinking Risk Management:
07.07.2009 - Featuring Chris Alberts

The Upside and Downside of Security in the Cloud:
06.16.2009 - Featuring Tim Mather

More Targeted, Sophisticated Attacks: Where to Pay Attention:
05.26.2009 - Featuring Marty Lindner

Is There Value in Identifying Software Security "Never Events?":
05.05.2009 - Featuring Robert Charette

Cyber Security, Safety, and Ethics for the Net Generation:
04.14.2009 - Featuring Rodney Petersen

An Experienced-Based Maturity Model for Software Security:
03.31.2009 - Featuring Gary McGraw

Mainstreaming Secure Coding Practices:
03.17.2009 - Featuring Robert Seacord

Security: A Key Enabler of Business Innovation:
03.03.2009 - Featuring Roland Cloutier

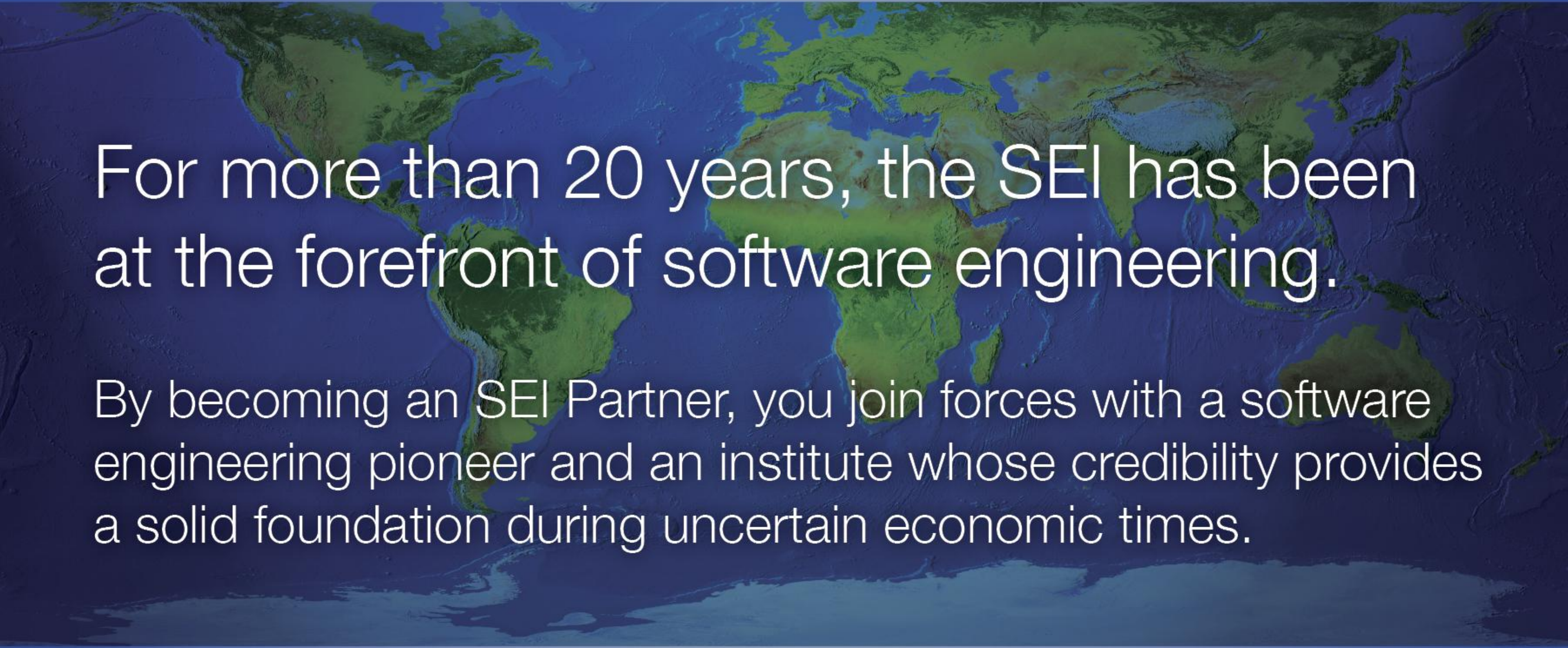
Better Incident Response Through Scenario Based Training:
02.17.2009 - Featuring Chris May

An Alternative to Risk Management for Information and Software Security
02.03.2009 - Featuring Brian Chess

Waiting for www.cert.org...

CERT's Podcast Series: Security for Business Leaders

▶ **www.cert.org/podcast/**



For more than 20 years, the SEI has been at the forefront of software engineering.

By becoming an SEI Partner, you join forces with a software engineering pioneer and an institute whose credibility provides a solid foundation during uncertain economic times.

SEI Partner Network

▶ www.sei.cmu.edu/partners

Are you interested in learning more? Visit http://www.sei.cmu.edu/architecture/saturn/ to

- Find out about the SEI software architecture work, current research, tools and practices, news, and how the SEI can help you.

- Stay connected to architecture experts through the SATURN Network on LinkedIn.

- Attend SATURN 2010 – the annual conference that brings together experts from around the world to exchange best practices in developing, acquiring, and maintaining software, systems, and enterprise architecture. Registration is now open!